fgets() and gets_s()

Robert C. Seacord, Software Engineering Institute [vita[1]]

2005-09-27

The `gets()` function is a common source of buffer overflow vulnerabilities and should never be used. The `fgets()` and `gets_s()` functions each offer a more secure solution.

Development Context

Reading strings from standard input

Technology Context

C, UNIX, Win32

Attacks

Attacker executes arbitrary code on machine with permissions of compromised process.

Risk

The `gets()` function is a common source of buffer overflow vulnerabilities and should never be used. Programs running with elevated privileges, including programs that are outward facing, can be used for privilege escalation or to launch a remote shell.

Description

There are two alternative functions that can be used: `fgets()` and `gets_s()`. Figure 1 shows how all three functions are used.

The `fgets()` function is defined in C99 [ISO/IEC 99] and has similar behavior to `gets()` . The `fgets()` function accepts two additional arguments: the number of characters to read and an input stream. By specifying stdin as the stream, `fgets()` can be used to simulate the behavior of `gets()` , as shown in lines 6-10 of Figure 1. The `fgets()` function, however, retains the new-line character, which means that the function cannot be used as a direct replacement for `gets()`.

The `fgets()` function reads at most one less than the number of characters specified from the stream into an array. No additional characters are read after a new-line character or after end-of-file. A null character is written immediately after the last character read into the array. The C99 standard does not define how `fgets()` behaves if the number of characters to read is specified as zero or if the pointer to the character array to be written to is a null.

The `gets_s()` function is defined by ISO/IEC WDTR 24731 to provide a compatible version of

1. daisy:274 (Seacord, Robert C.)

`gets()` that is less prone to buffer overflow. This function is closer to a direct replacement for the `gets()` function in that it reads only from the stream pointed to by stdin. The `gets()` function, however, accepts an additional argument of `rsize_t`. If this argument is equal to zero or greater than `RSIZE_MAX` or if the pointer to the character array to be written to is a null, then there is diagnosed undefined behavior, and no input is performed and the character array is not modified. Otherwise, the function reads, at most, one less than the number of characters specified, and a null character is written immediately after the last character read into the array. Lines 11-15 of Figure 1 show how `gets_s()` can be used in a program.

Figure 1. Use of gets() vs. fgets() vs. gets_s()

```
 1. #define BUFFSIZE 8
 2. int _tmain(int argc, _TCHAR* argv[]){
 3.   char buff[BUFFSIZE];
      // insecure use of gets()
 4.   gets(buff);
 5.   printf("gets: %s.\n", buff);
 6.   if (fgets(buff, BUFFSIZE, stdin) == NULL) {
 7.     printf("read error.\n");
 8.     abort();
 9.   }
10.   printf("fgets: %s.\n", buff);
11.   if (gets_s(buff, BUFFSIZE) == NULL) {
12.     printf("diagnosed undefined behavior.\n");
13.     abort();
14.   }
15.   printf("gets_s: %s.\n", buff);
16.   return 0;
17. }
```

The `gets_s()` function returns a pointer to the character array if successful. A `NULL` pointer is returned if the function arguments were invalid, an end-of-file is encountered and no characters have been read into the array, or if a read error occurs during the operation.

The `gets_s()` function only succeeds if it reads a complete line (that is, it reads a newline character). If a complete line cannot be read, the function returns NULL and sets the buffer to the null string. The function also clears the input stream to the next newline character.

The `fgets()` and `gets_s()` functions can still lead to buffer overflows if the specified number of characters to input exceeds the length of the destination buffer.

References

[ISO/IEC 99]	ISO/IEC. *ISO/IEC 9899 Second edition 1999-12-01 Programming languages — C*. International Organization for Standardization, 1999.
[ISO/IEC 04]	ISO/IEC. *ISO/IEC WDTR 24731 Specification for Secure C Library Functions*. International Organization for Standardization, 2004.

Pearson Education, Inc. Copyright

Velden

Naam	Waarde
Copyright Holder	Pearson Education

Velden

Naam	Waarde
is-content-area-overview	false
Content Areas	Knowledge/Coding Practices
SDLC Relevance	Implementation
Workflow State	Publishable